Computer Architecture

Some questions & answers

Prof. Nizamettin AYDIN, PhD

naydin@yildiz.edu.tr

http://www3.yildiz.edu.tr/~naydin

Q50

- MIPS processor was born in the early 1980s from the work done by John Hennessy and his students at Stanford University and is mainly used in small devices. So,
  - What architecture the MIPS processor adapted?
  - What are the advantages of this architecture?

A50

- MIPS adapted the RISC architecture to explore the architectural concept of RISC

- Advantages of RISC Architecture:
  - The performance of RISC processors is often two to four times than that of CISC processors because of simplified instruction set.
  - This architecture uses less chip space due to reduced instruction set.
  - The per-chip cost is reduced by this architecture that uses smaller chips consisting of more components on a single silicon wafer.
  - RISC processors can be designed more quickly than CISC processors due to its simple architecture.
  - The execution of instructions in RISC processors is high due to the use of many registers for holding and passing the instructions as compared to CISC processors.

Q51

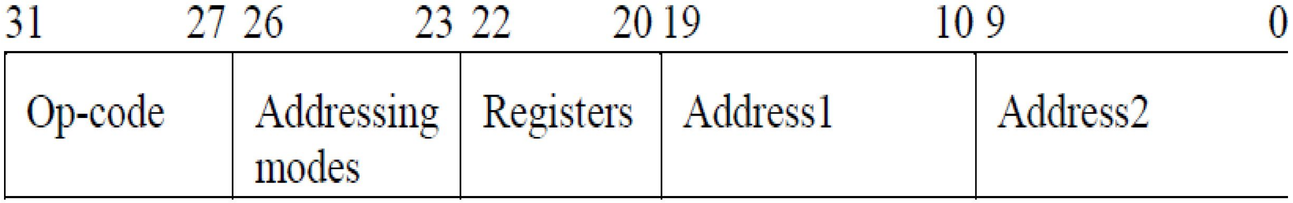
- List the characteristics of the Von Neumann computer model.

A51

- The Von Neumann computer model has the following characteristics:
  - Three hardware systems:
    - A central processing unit (CPU)
    - A main memory system
    - An I/O system
  - The capacity to carry out sequential instruction processing.
  - A single data path between the CPU and main memory.

Q52

- If a computer system, in which addressing is by 1 byte unit, has the following instruction format:



31 27	26 23	22 20	19 10	9 0
Op-code	Addressing modes	Registers	Address1	Address2

  - How many registers this computer system might have?
  - How many bits are needed for the program counter?
  - How many memory locations this instruction will occupy?
  - Total how many bits can be stored in the memory of this computer?

A52

- $2^3 = 8$ registers,
- 10 bits are needed
- 32 / 8 = 4 memory locations
- $2^{10} = 1024$ Byte = 1024×8 = 8192 bit,

Q53

- Data from a damaged file containing variables in 32 bit IEEE floating point number format needs to be recovered. One of the recovered data appears to be $(432AA000)_{fp(hex)}$.
  - What is the recovered data in decimal?
  - Find corresponding BCD code of the recovered data.

A53

- $(X_{10}=(-1)^{S}\times 1.M\times 2^{BE-127})$
  $(432AA000)_{fp(hex} = (01000011001010101010000000000000)_{fp(bin)}$
  $S \;=\; 0$
  $BE \;=\; (134)_{10} = (10000110)_2$ □ $E = (134-127)_{10} = (10000110 - 01111111)_2 = (00000111)_2$
  $M \;=\; 01010101010000000000000$
  $X_{10}=(-1)^{S}\times 1.M\times 2^{BE-127}) = (-1)^{0}\times 1.0101010101\times 2^{7} = 10101010.101 = (170.625)_{10}$

- $(170.625)_{10} = (0001\ 0111\ 0000.0110\ 0010\ 0101)_{BCD}$

Q54

- Consider the following assembly program written for a computer with a two-addres instruction processor. If a portion of the memory is given as beside (V, W, X, Y, and Z are variables);

```
100     MOVE    R1, (200)
101     SUB  R1, (201)
102     MOVE    R2, (202)
103     ADD R2, (204)
104     MPY     R1, R2
105     MOVE    (204), R1
```

Mem. Adress	Data
200	V = A4
201	W = A0
202	X = 72
203	Y = 22
204	Z = 41

Q54

- What function (in terms of the given variables) this program implements? (R1 and R2 are registers; MOVE: data movement instruction; SUB: subtraction instruction; MPY: multiplication instruction; ADD: addition instruction; DIV: division instruction)

- Assuming that V, W, X, Y, and Z are 2s complement signed numbers, what will be the values of *carry flag*, *sign flag*, *zero flag*, and *overflow flag* after the execution of instruction in memory location 103? Explain

A54

- Z = (V - W) (X + Y)

-
```
          1
72   □     0 1 1 1 0 0 1 0
41   □     0 1 0 0 0 0 0 1
B3   □   0 1 0 1 1 0 0 1 1
```

carry flag = 0 , sign flag = 1,

zero flag = 0, overflow flag = 1

Q55

- In the following MIPS code

```
lw      $t0,    32($s3)
add         $t0,    $s2,    $t0
sw      $t0,    48($s3)
```

  - How many times is instruction memory accessed?
  - How many times is data memory accessed?
  - How many times is register file accessed?
  - Which ones are read from the register file, which ones are write to the register file?

A55

- 3
- 2
- 7

- lw $t0, 32($s3) □ s0 is read from, t0 is written into

  add $t0, $s2, $t0 □ s2 and t0 are read from,
  t0 is written into

  sw $t0, 48($s3) □ s3 and t0 are read from

Q56

- The contents of memory and CPU registers of a computer system are given as following (the values are in hexadecimal).

Memory address	Memory content		Register Content	Register
10	**561**		**10**	**PC**
11	**160**			**AC**
12	**361**		**561**	**IR**
	.			
60	**003**			
61	**070**			
	.			
70	**005**			
71	**002**			

Q56

- What will be the content of the actual operand for the instruction being executed when the addressing mode is;
  - immediate addressing
  - direct addressing
  - indirect addressing
  - PC relative addressing

A56

- 061 (operand is adress)
- 070 (operand is the value in the address 61)
- 005 (operand is the value in the address 70 pointed by the address 61)
- 002 (operand is the value in the address calculated as 61+10 (content of PC))

Q57

- In an 8 bit microprocessor data transfer rate is given as 2 MByte per second.
- Find the clock frequency of the processor if a bus cycle takes 2 clock cycles.

A57

- If 2 MByte is transferred in 1 second, 1 Byte is transferred in **$1/(2\times10^{6}) = 0.5\times10^{-6}$ second** .

- If 1 byte requires 2 clock cycle to transfer, clock cycle is **$(0.5\times10^{-6})/2 = 0.25\times10^{-6}$** second.

- So, the clock frequency is **$1/(0.25\times10^{6}$ second) = 4 MHz**.

Q58

- Classify and briefly describe I/O methods that can be used in a computing system.

A58

- I/O can be controlled in four general ways.
  - Programmed I/O:
    - Reserves a register for each I/O device.
    - Each register is continually polled to detect data arrival.
  - Interrupt-Driven I/O:
    - Allows the CPU to do other things until I/O is requested.
  - Direct Memory Access (DMA):
    - Offloads I/O processing to a special-purpose chip that takes care of the details.
  - Channel I/O:
    - Uses dedicated I/O processors.

Q59

- Suppose a system has a virtual memory size of 64K and the system uses word addressing.

- If the page size is 1024 Word,
  - How many bits this virtual memory address requires?
  - Determine the number of virtual pages

A59

- virtual address space = 64K = $2^6 \times 2^{10}$ = 2^{16} Word
- page size = 1024 Word = 2^{10} Word
  - Because the virtual memory size is 2^{16} Word, the virtual memory address requires **16 bits**
  - Number of virtual pages = $2^{16}/2^{10}$ = 2^{16-10} = 2^6 = **64**

Q60

- Assume that you have a computer with a maximum main memory size of 2 GByte (motherboard does not allow to upgrade the main memory).
- The program you want to run on this computer may require a larger main memory, resulting in a virtual memory that uses part of the hard drive.
  - However, you realized that the magnetic hard drive in this computer is too slow for your application.
- What can you do to increase the performance of this computer?
- Explain how and why your solution will increase the performance.

A60

- I would replace magnetic hard drive with a Solid State Drive (SSD).

- SSDs have the following advantages over HDDs:
  - High-performance input/output operations per second (IOPS),
  - less susceptible to physical shock and vibration longer lifespan,
  - lower power consumption,
  - lower access times and latency rates.

Q61

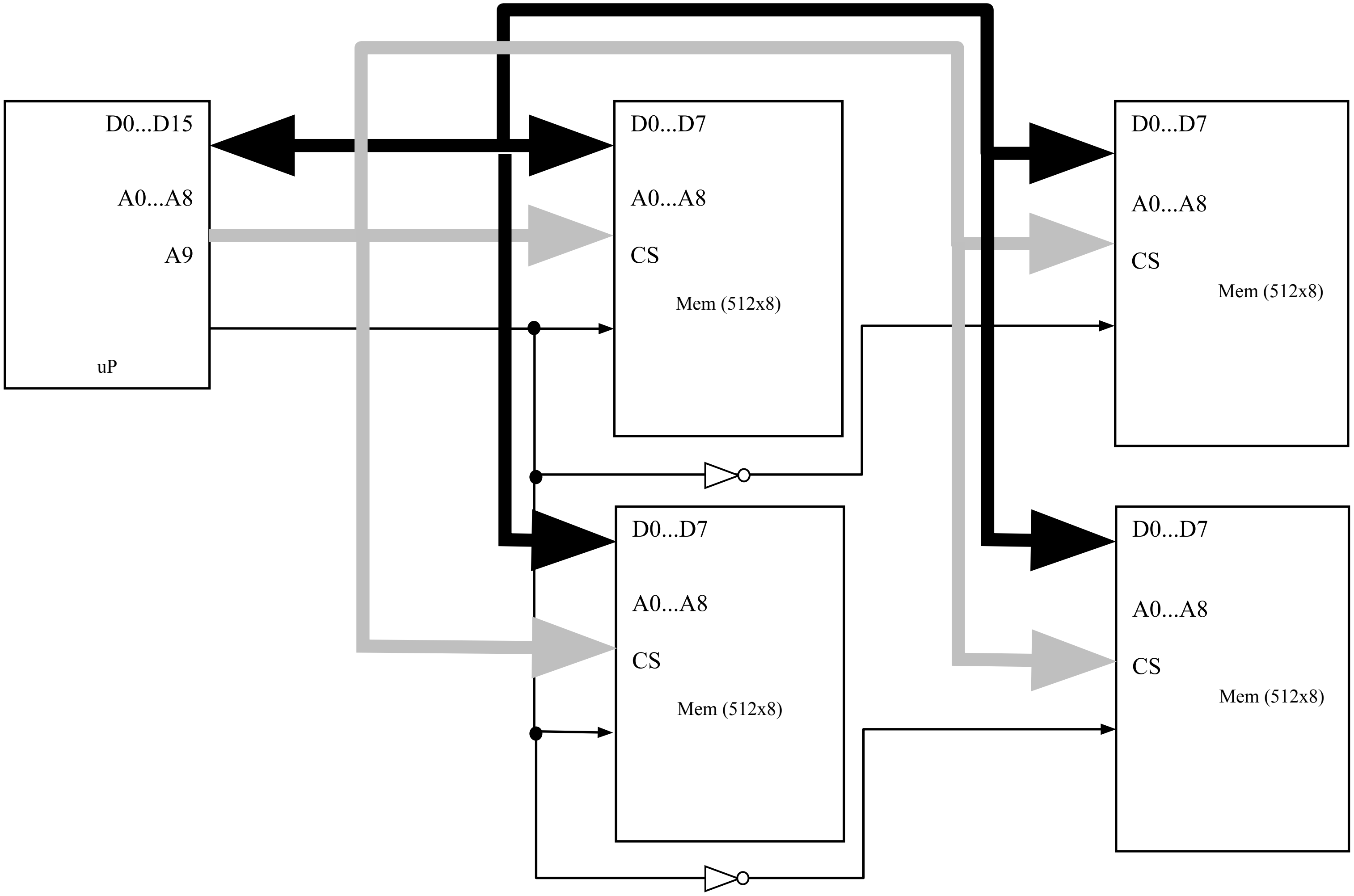
- A 16 bit microprocessor system has 10 address lines to address relevant memory locations.
  - Assuming that the data size is 1 byte, what is the address of the last memory location?
  - Assuming memory chips organized as 512x8 bits are used, how many memory chips we need to design the required memory).
  - Design the required memory system using the memory chips organized as 512x8 bits.

A61

- The last memory location
  = $2^{\#\text{of bits in address line}} - 1 = 2^{10} - 1 = 1023$

- Data bus size **16 bits**, memory size is **1024**. System memory organization is **1024x16**.
- If we restate this memory organization in terms of the available memory chip (512x8), we have {(2x512)x(2x8)} = **4**x(512x8).
  - So, We need **4** (512x8) memory chips to have a memory organized as 1024x16

A61


D0...D15
A0...A8
A9
uP
D0...D7
A0...A8
CS
Mem (512x8)
D0...D7
A0...A8
CS
Mem (512x8)
D0...D7
A0...A8
CS
Mem (512x8)
D0...D7
A0...A8
CS
Mem (512x8)

Q62

- Consider a processor which uses branch delay technique for optimization of pipelining. Each intruction has 2 or 3 stages depending on the instruction type (I: Instruction fetch, E: Execute D: Memory operation)

```
100 LOAD     R1, (500)
101 JUMP     105
102 ADD      R1, 1
103  AND     R1, (501)
```

  - Which instruction will be executed first after the processor encounters the instruction in line 101.
  - Draw the relevant pipelinening diagram for the given program.

A62

- First, instruction in line 100, which adds contents of memory location 500 to R1 will be executed.

- In line 101 there is a jump (unconditional branch) instruction.
  - Because the processor uses a branch delay technique, instruction in line 102 will be executed before processor jumps to line 105.

A62

	1	2	3	4	5	6
100 LOAD R1, (500)	I	E	D			
101 JUMP 105		I	E			
102 ADD R1, 1			I	E		
103 AND R1, (501)				I	E	D

Q63

- Why a cash and virtual memory are used in a computing system?
  - Briefly explain.

A63

- Cache memory enhances performance by providing faster memory access speed.
  - The purpose of cache memory to speed up accesses by storing recently used data closer to the CPU.
  - It is much smaller than main memory. Its access time is a fraction of that of main memory.
- Virtual memory enhances performance by providing greater memory capacity, without the expense of adding main memory.
  - Instead, a portion of a disk drive serves as an extension of main memory.

Q64

- Suppose we have 32-bit memory addresses, a byte-addressable memory, and a 512 KB cache with 32 bytes per block.

1. How many total lines are in the cache?
2. If the cache is direct-mapped, how many cache lines could a specific memory block be mapped to?
3. If the cache is direct-mapped, what would be the format (Tag-id, Line-id, Word-id) of the address? (Clearly indicate the number of bits in each)

Q64

4. If the cache is 2-way set associative (sets of size 2), how many cache lines could a specific memory block be mapped to?
5. If the cache is 2-way set associative (sets of size 2), how many sets would there be?
6. If the cache is 2-way set associative, what would be the format of the address?
7. If the cache is fully-associative, how many cache lines could a specific memory block be mapped to?
8. If the cache is fully-associative, what would be the format of the address?

A64

1. $\frac{512\text{ KBytes}}{32\text{ Bytes}} = \frac{2^{19}}{2^{5}} = 2^{14}\text{ lines} = 2^{4} \times 2^{10}\text{lines} = 16\text{ K lines} = 16384\text{ lines}$

2. **1** (a specific memory block could be mapped to only 1 line in the cache)

3. Each block contains 32 bytes. Therefore, 5 bits are needed to specify the *Word-id*.

   There are 2^{14} lines in the cache. Therefore, 14 bits are needed to specify the *Line-id*.

   Tag-id bits in this case is defined as *Main memory address* – (*Line-id bits* + *Word-id bits*). Therefore, 32 - (14 + 5) = 13 bits are needed to specify the *Tag-id*.

   The format of the address:

	Tag-id	**Line-id**	**Word-id**
Main memory address =	**13 bits**	**14 bits**	**5 bits**

A64

4. **2** (a specific memory block could be mapped to any 2 lines within 1 set in the cache)

5. $$\frac{\mathbf{2^{14}\ lines}}{\mathbf{2^{1}\ lines}} = \mathbf{2^{13}\ sets} = \mathbf{2^{3} \times 2^{10} sets} = \mathbf{8\ K\ sets} = \mathbf{80192\ sets}$$

6. Each block contains 32 bytes. Therefore, 5 bits are needed to specify the *Word-id*.
There are 2^{13} sets in the cache. Therefore, 13 bits are needed to specify the *Set-id*.

Tag-id bits in this case is defined as *Main memory address* – (*Set-id bits* + *Word-id bits*). Therefore, 32 - (13 + 5) = 14 bits are needed to specify the *Tag-id*.

	Tag-id	Set-id	Word-id
Main memory address =	**14 bits**	**13 bits**	**5 bits**

A64

7. **2^{14}** (a specific memory block could be mapped to any cash line in the cache)

8. Each block contains 32 bytes. Therefore, 5 bits are needed to specify the *Word-id*.

Tag-id bits in this case is defined as *Main memory address – Word-id bits*. Therefore, 32 - 5 = 27 bits are needed to specify the *Tag-id*.

	Tag-id	**Word-id**
Main memory address =	**27 bits**	**5 bits**